NBA Salary Prediction and Assessment

Problem Statement

As the world's most elite basketball athletes, NBA players in general receive high income as a result of their talent and hard work. Most of the time, NBA players who are under the final year of their contracts have to have good performances along with good looking statistics (points, rebounds, etc.) for them to receive big contracts the next year. Certain players could have received a long term big contract (4 years is deemed to be a long term contract), but they might not perform well after they got paid. The purpose of this project is to create a model that will help us predict a player's salary in the 2021-2022 season with their respective stats from 2020-2021 season. Moreover, as some of the players have received their contracts from years back, this model could also serve as a yardstick to evaluate a player's performance to see whether he is being overpaid or underpaid.

I. Data Collection

The data for this project is collected from website Pro Basketball Reference, and the specific dataset being collected are:

- 2020-2021 NBA Player Stats: Per Game
- 2020-2021 NBA Player Stats: Advanced
- NBA Contracts Summary

II. Data Wrangling

Since the datasets are collected from the same website, I simply merged the three datasets with the player's unique ID. One of the problems I encountered in merging these datasets is that certain players were traded during the season, as a result, they have different stats for different teams. I decided to keep players' stats for each team, since maybe a player might start performing well after getting traded, resulting in receiving a bigger contract.

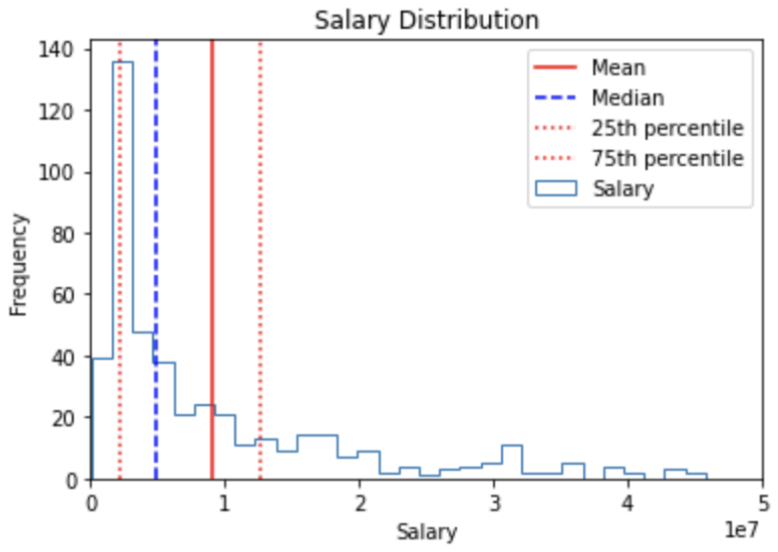
Moreover, I dropped the players that have missing values for FG% (field goal percentage), since those players might have never shot a ball once on their respective team. After dropping those players, I kept the players with a missing 3 point FG%, since those players might have received that result from not shooting a 3 the whole season, but they still played a good amount of time that season.

I wanted to keep as many observations as possible, since there are generally not a lot of players in the league. Consequently, the shape of our dataset has been reduced from 1221 rows x 58 columns to 454 rows * 51 columns (After deleting duplicate rows and players that did not play much).

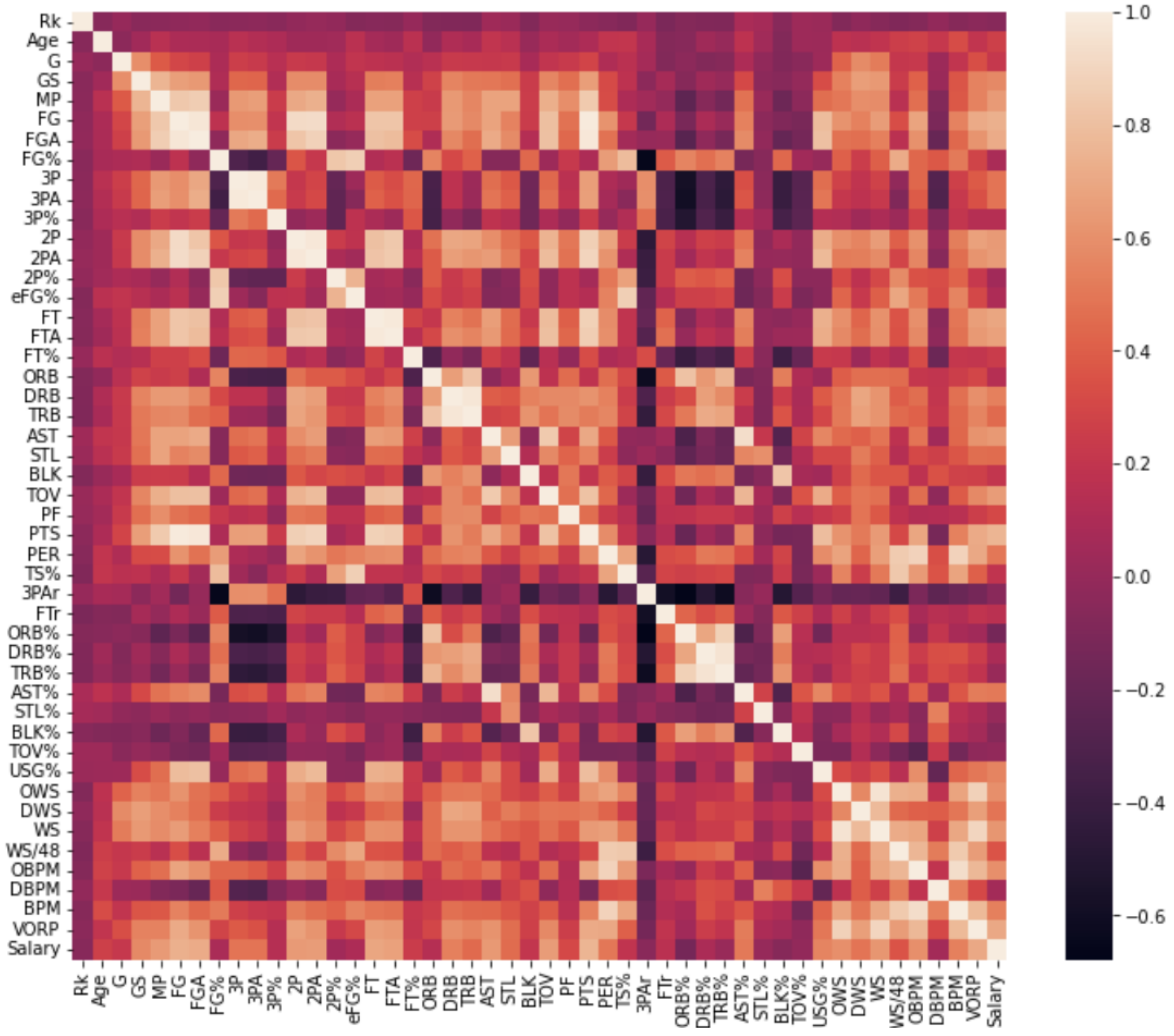
III. Exploratory Data Analysis

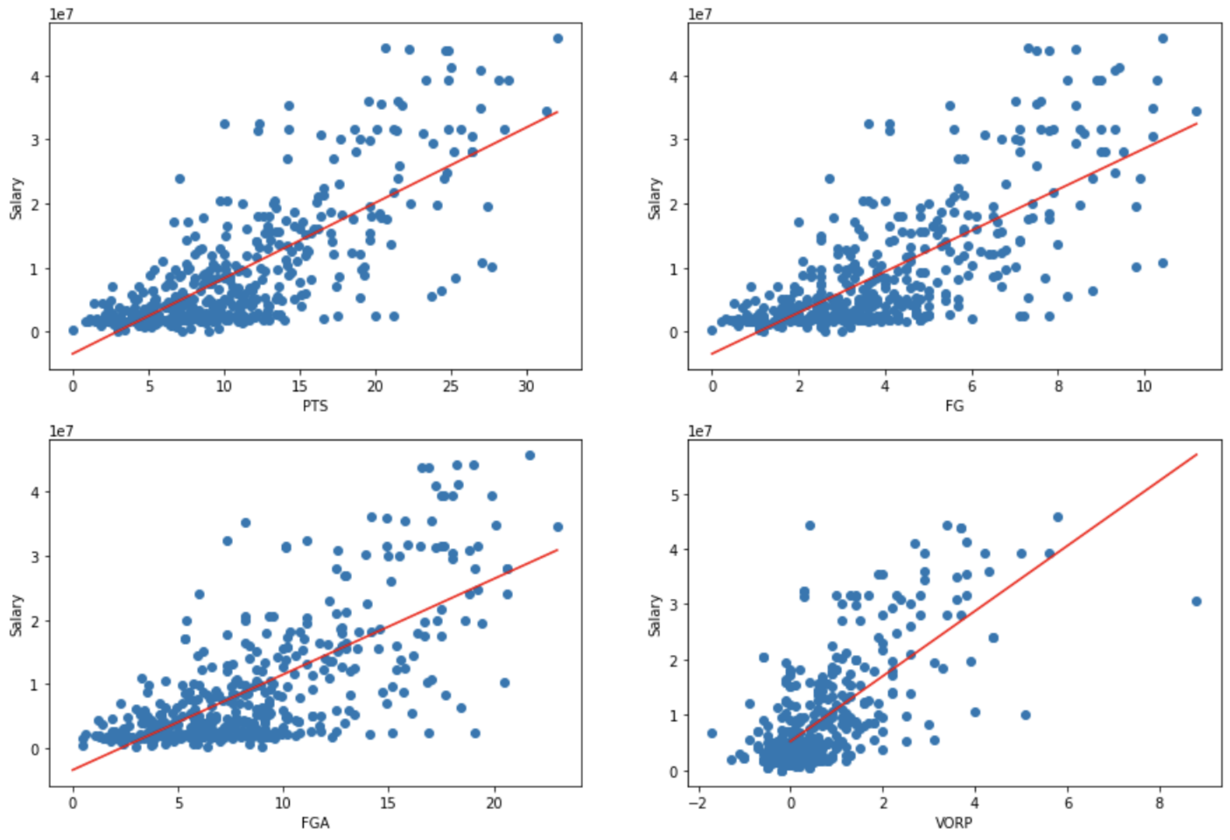
From EDA, there are a few interesting stats:

- The average point scored by NBA players is around 10.65.
- The NBA is a young players driven league, with players under 25 occupying more than 50% of the league roster spots.
- While shooting percentages and 3 point percentages tend to follow a normal distribution, field goals made and attempts tend to be a shape that is skewed to the right.
- Salary is also skewed to the right, with certain bins popping at the very end

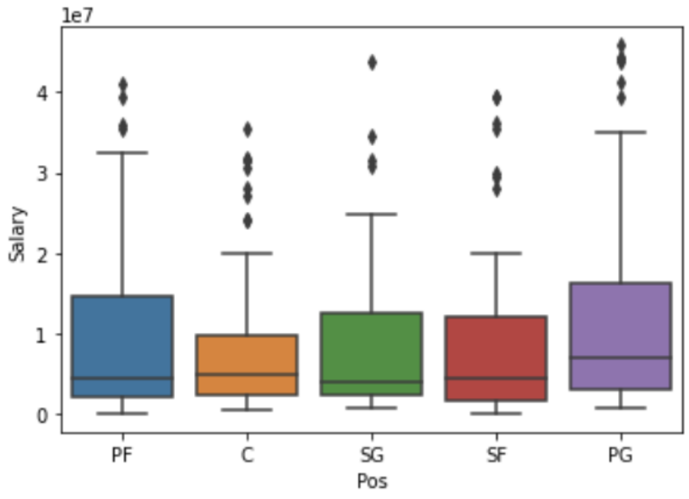


- Points, FG makes, FG attempts, and VORP (Value over replacement players) have the highest correlations with salary





For our exploratory analysis, we conducted a permutation test to see whether we need to remove categorical variable – positions



As we can see here that PG has a higher mean for salary than the rest of the positions (Positions except PG have similar means for salary).

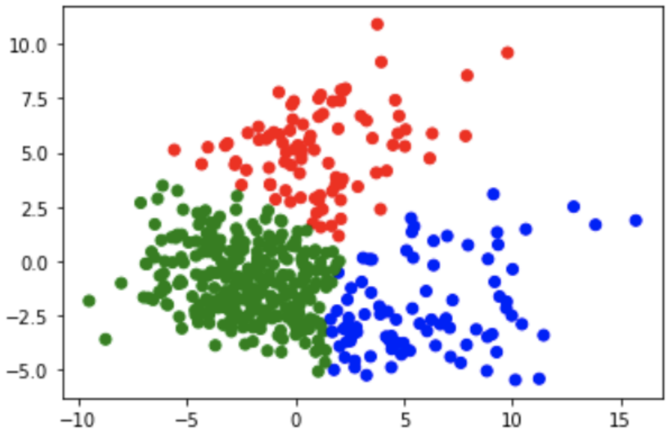
Null hypothesis: Being point guards do not have any impact on a player's salary. We reject the hypothesis with p<0.05.

After our test, we received a P-value lower than 0.05, rejecting the null hypothesis and kept the positions for each player.

IV. Preprocessing and Training Data

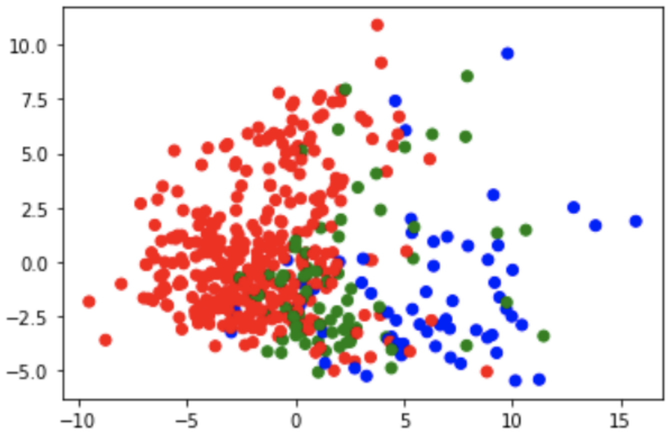
For feature engineering, we filled the Null values for 3 point percentage with 0, indicating that the player has a 0% of making a 3, since they did not attempt a single 3 point shot. Moreover, we made Position into dummified data. As a result, we have our final dataset before modeling, with 454 rows and 53 columns.

Moreover, I used principal component analysis (with 2 components) to see whether we can make clusters for players with different salary levels (with threshold being 10M, 20M), below is the result of our PCA. The last graph is the actual scatterplot with the players' salary levels.



```
plt.scatter(x,y,c=df['Status'], cmap=clp(['red', 'green', 'blue']))
```

```
<matplotlib.collections.PathCollection at 0x7f97e1bed880>
```



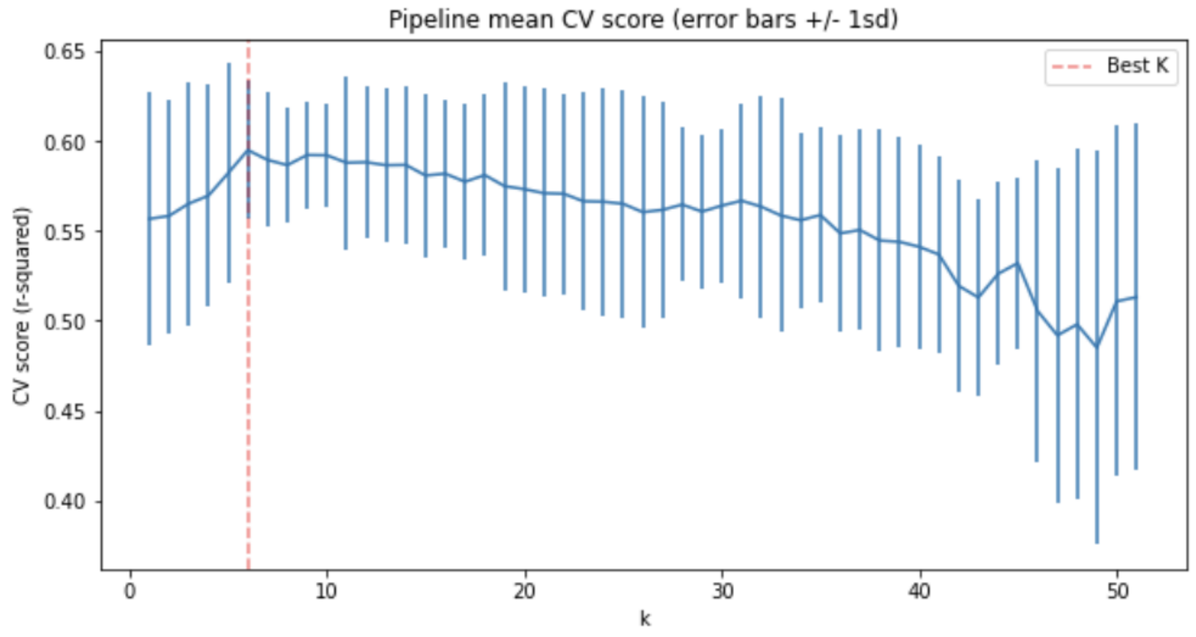
V. Modeling

For our modeling part, we decided to use three regression models to help us make our final prediction.

- Linear Regression

For our first model, we did a linear regression model, with parameters Standard Scaler and Select K best. Below is the CV score after grid searching.

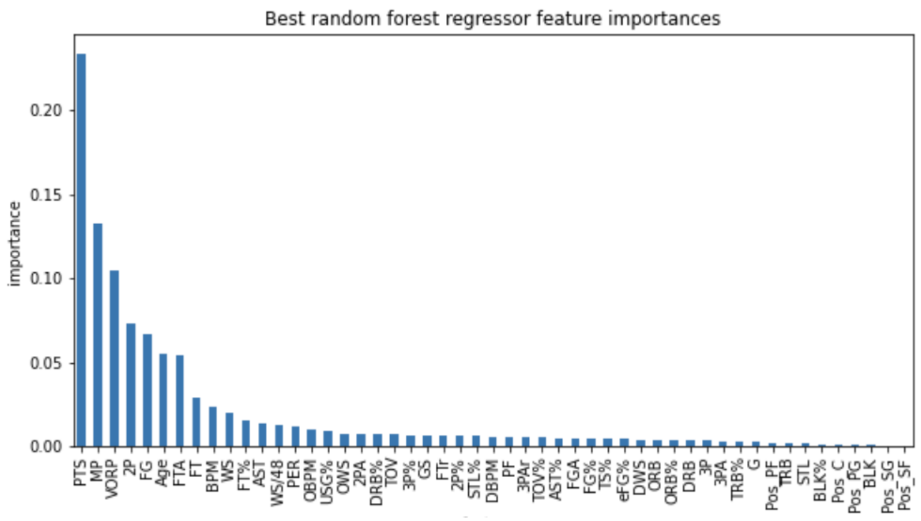
The four most important features: PTS, VORP, FGA, FT.



- Random Forest Regression

For our random forest regressor, we have hyperparameters like standard scaler, max depth, bootstrap, minimum sample leaf, and n_estimator. Below is our best parameter with the best training performance (With CV score = 0.6518).

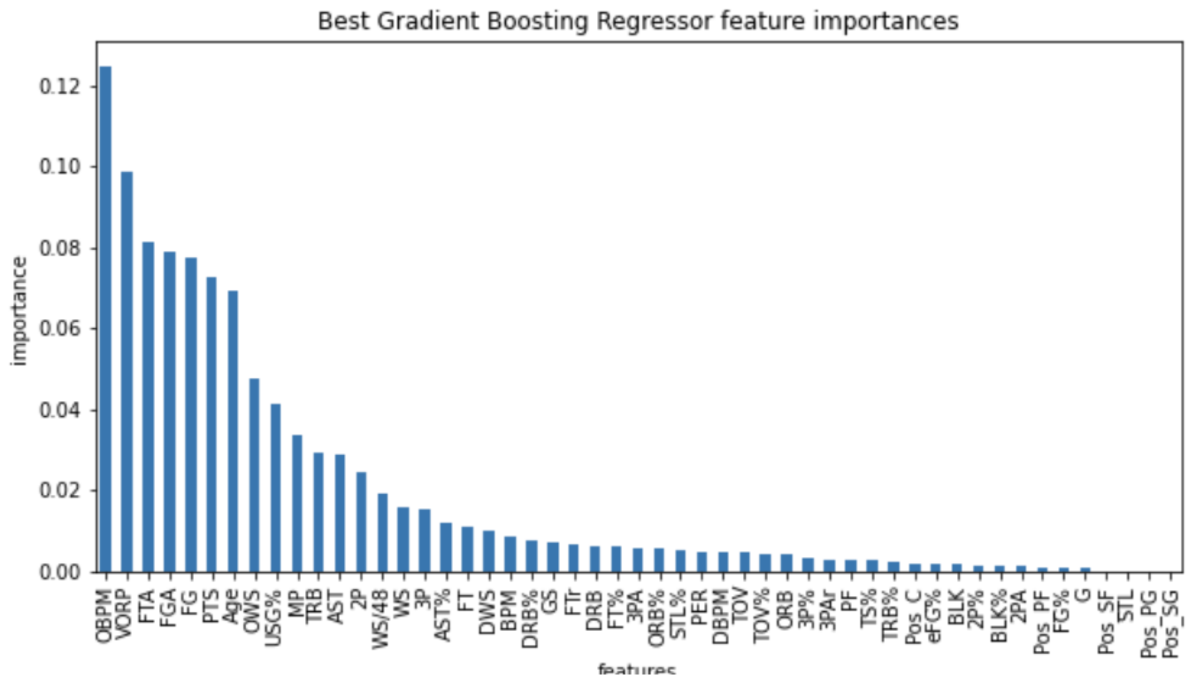
‘randomforestregressor__bootstrap': True,
'randomforestregressor__max_depth': 80,
'randomforestregressor__min_samples_leaf': 3,
'randomforestregressor__n_estimators': 33,
'standardscaler': StandardScaler()



- Gradient Boosting Regressor

Below are the hyperparameters for our gradient boosting regressor.

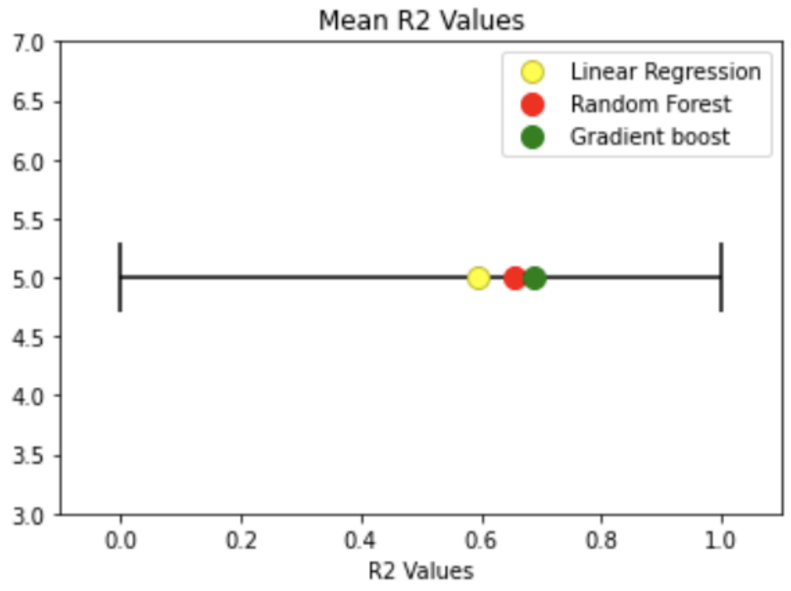
```
'standardscaler': [StandardScaler(), None],
   "gradientboostingregressor__max_features": [3,4,5,6,7],
   "gradientboostingregressor__max_depth": [3,4,5],
   "gradientboostingregressor__n_estimators": n_est,
   "gradientboostingregressor__learning_rate": [0.001,0.01,0.1]
```



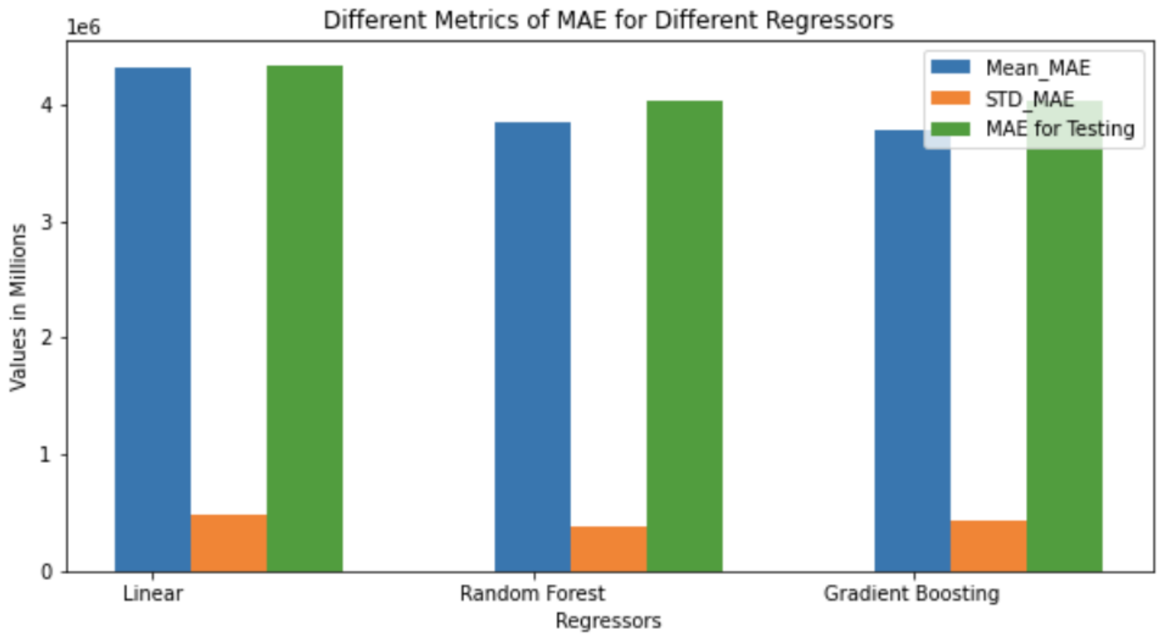
Final Assessment

For assessment, I used R squared, MAE, and RMSE as our metrics.

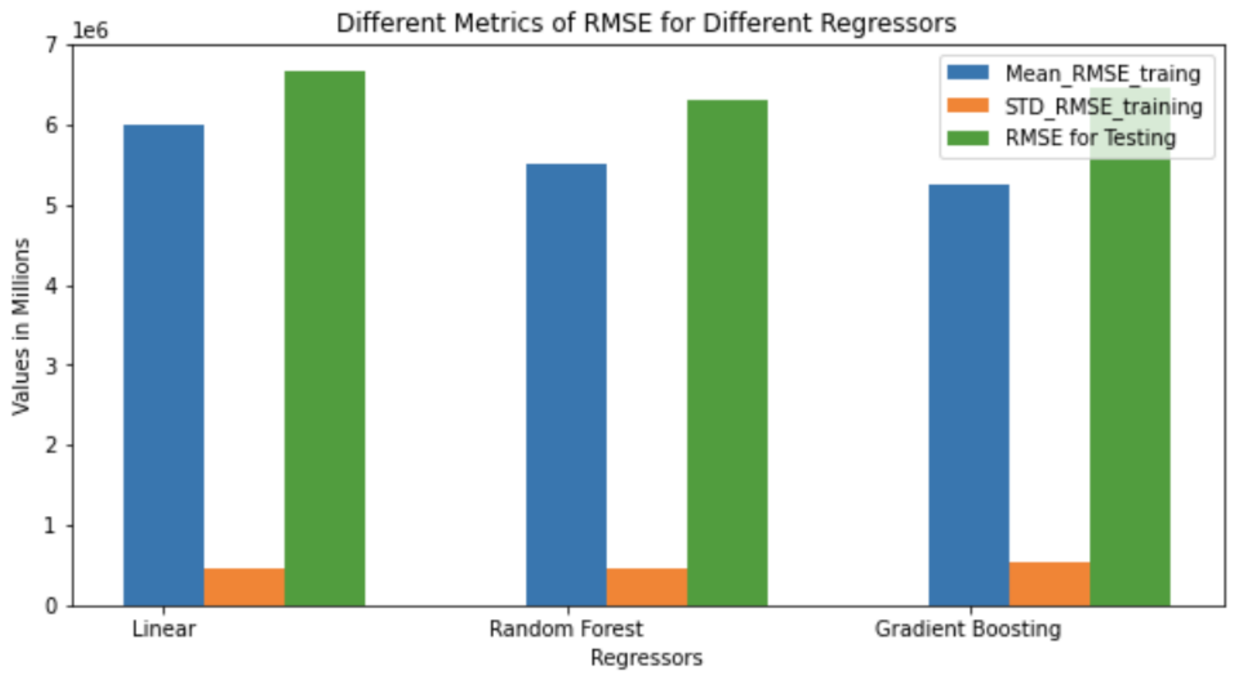
- R squared



- MAE



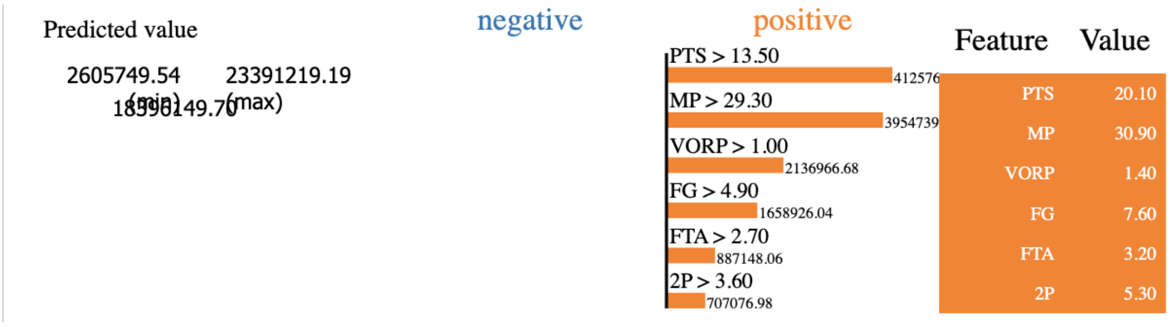
- RMSE



Final Model Selection and Reasoning

From the metrics we have chosen, it seems like Random Forest Regressor with its respective best parameters would outperform the other two models. It has the lowest mean and std for mean absolute error of the training set, and the mean absolute of the prediction from the testing data is the lowest among the three models as well. Moreover, random forest regressor happens to perform well on the test set compared to gradient boosting regressor, which has a lower value for the RMSE when it comes to training data, indicating that gradient boosting regressor could be overfitting.

Below is a LIME interpretation of our model with the player being selected being Kristaps Porzingas. From our model, it seems like Porzingas is a little overpaid for the 2021-2022 season with his performance from the 2020-2021 season.



Feature	Value
PTS	20.10
MP	30.90
VORP	1.40
FG	7.60
FTA	3.20
2P	5.30

```
Kristaps_Porzingas = df[df.Player==players_test.iloc[6,0]]
```

```
Kristaps_Porzingas.Salary
```

```
346    31650600
Name: Salary, dtype: int64
```